Structured Grid Load Decomposition
SGLD: A Short User's Guide

Xudong Xiao
Corvid Technologies, Inc.
145 Overhill Drive
Mooresville, NC 28117

July 13, 2007

Contents

List of Figures

Chapter 1

Installation

The directory tree of the split/merge utility (SGLD) is as follows:

```
SGLD +
     +--doc
     +--include
     +--metis-4.0  (optional)
     +--src
     +--sample
     +--vulcan
```

The doc directory contains the documentation relative to SGLD. The include directory contains the commnd.h and common.h files and the *.mod files generated by compiling the Fortran 90 modules in the src and vulcan directories. The vulcan directory contains the VULCAN source codes required for parsing the VULCAN input file. All the source codes developed for splitting the grid, input deck, and restart files are in the src directory. The source codes developed for merging restart files and the PLOT3D files created by the VULCAN post-processor are also located in the src directory. Finally, the sample directory contains several sample cases that illustrate how to use the SGLD software.

The installation of the SGLD software is performed as part of the VULCAN install process. This utility is installed when any of the following install options are invoked:

```
install_vulcan new
install_vulcan all
install_vulcan utl
```

The install process will result in four executable files in the src directory, namely, grid_split, restart_split, restart_merge, and plot3d_merge. The function of each executable is listed below:

Executables	**Function**
grid_split	split the grid and input deck
restart_split	split VULCAN restart files
restart_merge	merge VULCAN restart files
plot3d_merge	merge PLOT3D post-processing files

The SGLD utility also allows the option to utilize the METIS partitioning software to determine the mapping of grid blocks to processors. If this functionality is desired, then download the METIS software (Version 4.0):

```
http://glaros.dtc.umn.edu/gkhome/metis/metis/download
```

and install it prior to enabling the VULCAN_METIS environment variable in the VULCAN configuration file (vulcan_config). The METIS installation directory (metis-4.0) must be placed in the folder that contains the SGLD utility software, *i.e.*:

```
vulcan_install_path/Ver_X.X/Utilities/Load_Balance_codes/SGLD
```

1.1 Source Code Description

Source Code	**Description**
`grid_split.f90`	main program driving the splitting process
`block_vul.F`	1) collect grid, boundary condition, cut, and patch information 2) split the grid blocks 3) generate the new VULCAN input file 4) generate the new grid file 5) generate the block mapping file for restart file merging/splitting
`read_constraint.F`	reads the constraint file
`parser.f90`	parses strings
`block.f90`	defines data structures and performs basic operations on them
`block_merge.f90`	merges blocks and block dependent data constructs
`block_out.f90`	1) screen output of splitting results 2) laminar/ignition/time history sub-blocks calculation and output
`restart_split`	main program driving VULCAN restart file splitting
`restart_merge`	main program driving VULCAN restart file merging
`rstms.f90`	routines for restart merging and splitting
`plot3d_merge`	main program driving PLOT3D file merging

Chapter 2

Grid Splitter

2.1 Input Files

Three input files are required to execute the splitting process. They are:

1. the VULCAN input file
2. the PLOT3D (or PLOT2D) format grid file (file name specified in the VULCAN input file)
3. the SGLD input file (`grid_split.inp`), which contains the information that drives how the grid is to be split (the syntax and other details for this file are described in Section 2.2)

2.2 SGLD Input File

The split process control file, `grid_split.inp`, consists of multiple sections: the mandatory GLOBAL section, and the optional BLOCK and BC sections. The GLOBAL section *must* be the first section in the file. Each section starts with "`START` *section_name*" and ends with "`END`". The end of the entire file is indicated by "`EOF`". An example is given below:

```
START GLOBAL
  SPLIT_OPT         1
  NUM_PROCS         8
  LOAD_PERC        95
  NUM_LEVELS        2
  BLK_MIN_DIM       4
  OLD_VULCAN_INPUT  vulcan.inp
  NEW_VULCAN_INPUT  split.inp
  NEW_GRID          split.grd
END
START BLOCK
#BLOCK   KEYWORD  AXIS  BEG  END
   1    NO_SPLIT    I    29   33
   2      SPLIT     J    17
END
```

```
START BC
#BC NAME   KEYWORD
  INFLOW   NOT_ALLOWED
END
EOF
```

Blank lines are ignored, and a comment line is indicated by "#" as the first non-blank character. The keywords for each section are described below.

2.2.1 GLOBAL Section

The GLOBAL section is defined as:

```
START GLOBAL
  KEYWORD   VALUE
  ...
END
```

and this section can contain any subset of the following keywords:

Keyword	**Type**	**Description**
SPLIT_OPT	integer	indicates the algorithm used for splitting 0– performs a split at user specified index locations as specified in the BLOCK section of the SGLD input file 1– attempts to divide the grid into large "chunks", if the desired load balance criteria is not met, the size of the "chunk" is iteratively reduced until either the load balance criteria is met, or all blocks are of size BLK_MIN_DIM 2– divides the grid into small "chunks" based on the BLK_MIN_DIM parameter specified in the SGLD input deck
NUM_PROCS	integer	number of processors to load balance over
LOAD_PREC	integer	indicates the load balance tolerance (specified as a percentage) – typical values range between 95 and 100
BLK_MIN_DIM	integer	minimum cell dimension for a block
NUM_LEVELS	integer	number of multi-grid levels
OLD_VULCAN_INPUT	string	name of the VULCAN input file to split
NEW_VULCAN_INPUT	string	name of the new split VULCAN input file
NEW_GRID	string	grid file name for the new split grid

`METIS_OPT`	integer	indicates how to partition the graph 0– partitions the graph based on computation cost in a manner consistent with the default approach used by VULCAN **(this is the only option if METIS has not been downloaded and installed in the SGLD parent directory)** 1– calls METIS_PartGraphKway; partitions the graph into k equal size parts using the multi-level k-way partitioning algorithm (*i.e.* KMETIS with the objective of minimizing edgecut) 2– calls METIS_PartGraphVKway; partitions the graph into k equal size parts using the multi-level k-way partitioning algorithm (*i.e.* KMETIS with the objective of minimizing total communication volume) 3– calls METIS_PartGraphRecursive; partitions the graph into k equal size parts using a multi-level recursive bisection algorithm (*i.e.* PMETIS with the objective of minimizing edgecut)
`METIS_WGT_FLAG`	integer	indicates how the graph is weighted by METIS: 1– weights on edge/communication cost only 2– weights on vertex/computation cost only 3– weights on both vertices and edges
`SAME_CPU_MERGE`	string	indicates if blocks should be merged back together on each CPU Y– attempt to merge blocks N– do not merge blocks

2.2.2 BLOCK/BC Section

The BLOCK/BC sections share the same structure:

```
START BLOCK/BC
  BLOCK/BC  KEYWORD  [value1,  [value2, value3 ...]]
  ...
END
```

Each entry of this section begins with the block number (for blocks), or a BC name (for BCs), followed by a keyword. Depending on the keyword chosen, a different number of parameters may follow. Note that a BC type can also be specified in lieu of a BC name. The available block specific constraint options include:

Keyword	**Type**	**Description**
`SPLIT`	2 integers	the index at which to force a split format: `axis  index` where axis = 1, 2, 3 corresponds to I-, J-, K-axis, respectively (I, J, K can also be used to choose the axis) splitting is performed at the chosen index location m

`NO_SPLIT`	3 integers	the index range where no splitting is allowed format: `axis  start_index  end_index` where axis = 1, 2, 3 corresponds to I-, J-, K-axis, respectively (I, J, K can also be used to choose the axis) no splitting is allowed at location m, if $m_{beg} < m < m_{end}$
`MIN_DIM`	integer	minimum cell dimension for BLOCK n
`NOT_ALLOWED`	integer	no splitting is allowed for BLOCK n

The only supported BC specific constraint is `NOT_ALLOWED`:

Keyword	**Type**	**Description**
`NOT_ALLOWED`	character	no splitting is allowed for this BC name (or type)

This constraint is required for boundary conditions that utilize profile files, since the profile files are not included as part of the split process. The SGLD utility will also not allow a split to intersect a patch boundary. This is the default behavior of SGLD, so no constraint is required for this action.

2.3 Examples

2.3.1 Three Dimensional Duct

An initial two-block grid for a three dimension duct is shown in Fig. 2.1. Two different sample constraint files are described to show the capabilities of the grid splitter. In the first constraint file only a GLOBAL section is defined, while in the second one, constraints are defined in the GLOBAL, BLOCK, and BC sections. The first split control file is the following:

```
START GLOBAL
  SPLIT_OPT         2
  NUM_LEVELS        1
  BLK_MIN_DIM       4
  OLD_VULCAN_INPUT  injector.inp
  NEW_VULCAN_INPUT  split.inp
  NEW_GRID          split.grd
END
EOF
```

and the second constraint file is given by

```
START GLOBAL
  SPLIT_OPT         2
  NUM_LEVELS        2
  BLK_MIN_DIM       4
  OLD_VULCAN_INPUT  injector.inp
  NEW_VULCAN_INPUT  split.inp
  NEW_GRID          split.grd
END
```

```
START BLOCK
#BLOCK   KEYWORD  AXIS  BEG  END
   1    NO_SPLIT    I    30   40
END

START BC
#BC NAME   KEYWORD
  INFLOW   NOT_ALLOWED
END
EOF
```

The grids resulting from the first and second constraint files are shown in Fig. 2.2 and Fig. 2.3, respectively.

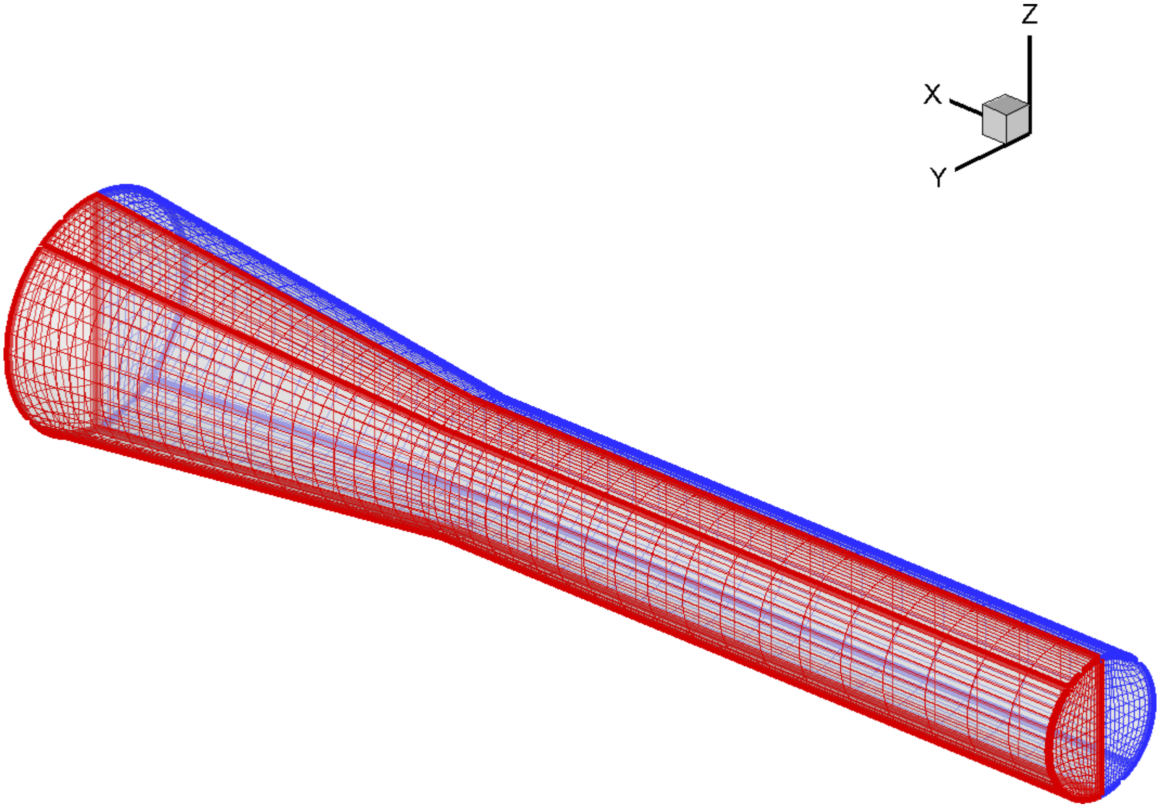

Figure 2.1: Initial 2-block grid, 3D duct

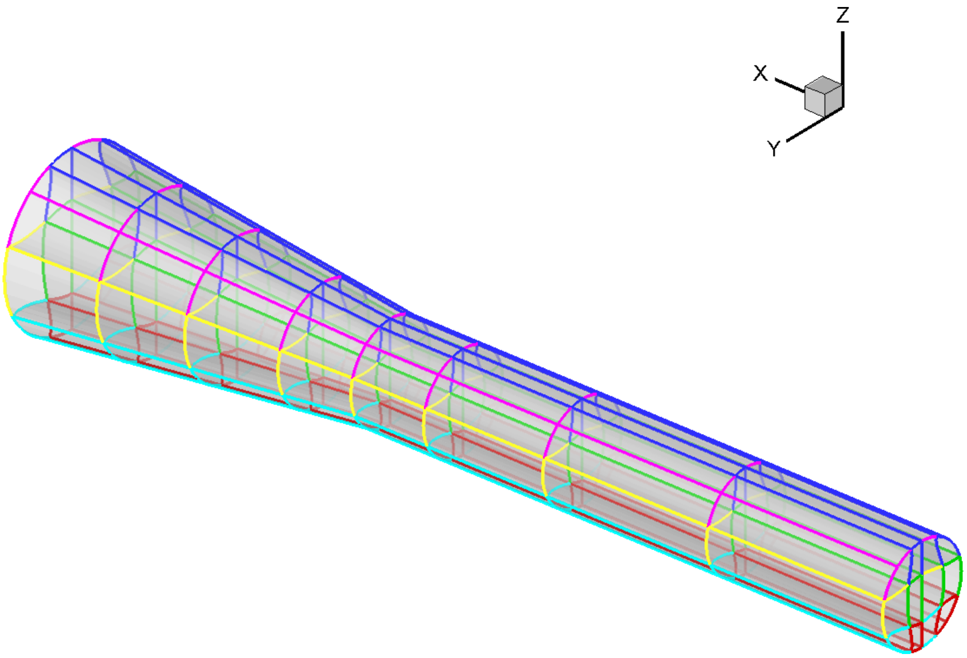

Figure 2.2: 96-block grid due to global constraints only

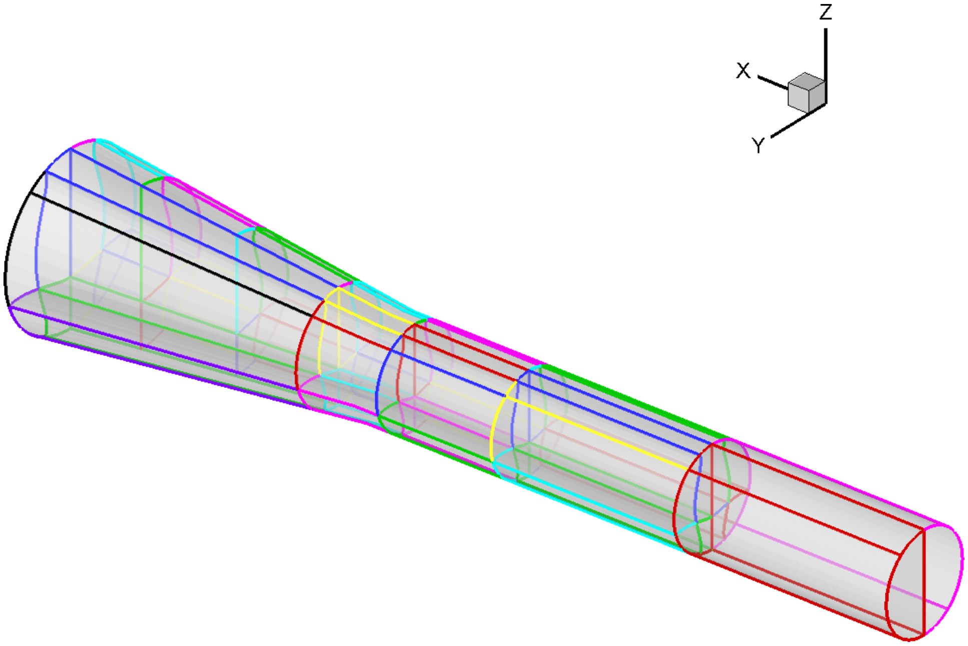

Figure 2.3: 42-block grid due to GLOBAL/BLOCK/BC constraints

2.3.2 Two Dimension Inlet

An initial eight-block grid for a two-dimensional hypersonic inlet is shown in Fig. 2.4. Using the following constraint file:

```
START GLOBAL
  SPLIT_OPT         2
  NUM_LEVELS        2
  BLK_MIN_DIM       20
  OLD_VULCAN_INPUT  inlet.input
  NEW_VULCAN_INPUT  split.input
  NEW_GRID          split.grd
END
EOF
```

results in the split grid shown in Fig. 2.5.

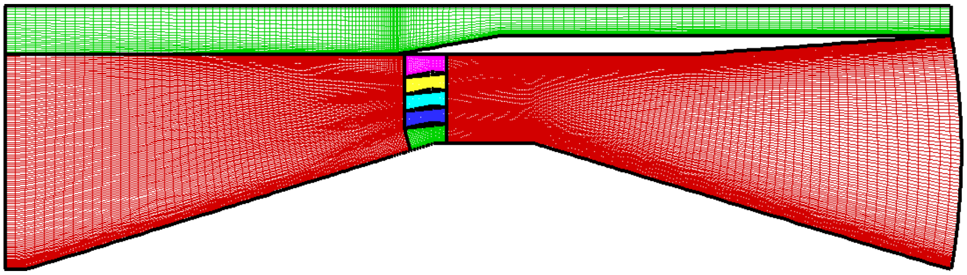

Figure 2.4: Initial 8-block grid, 2D inlet

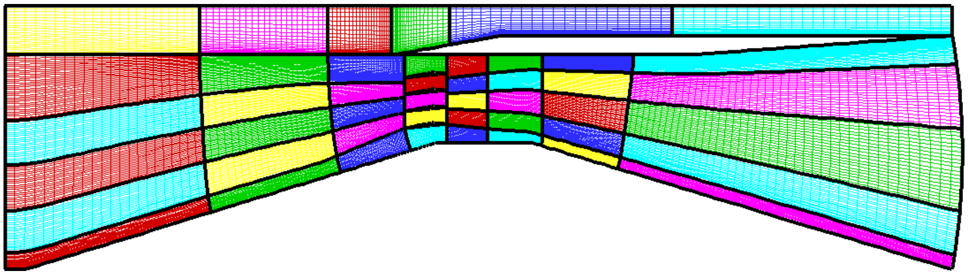

Figure 2.5: 46-block grid, 2D inlet

Chapter 3

Splitting Restart Files

The **grid_split** utility of SGLD splits the VULCAN input deck and the grid file. However, this utility does not yet have the capability to maintain specified initializations (propagation flags) that may have been used in the IN-ORDER column of the BC/CUT/PCH sections of the VULCAN input deck. As a result, all propagation values are zeroed out in the split input deck, preventing the desired initialization from being performed. An effective work-around for this issue is execute VULCAN using the original (non-split) input deck with the number of iterations set to zero. This will create a set of restart files that contain the desired initialization. The **restart_split** utility of SGLD can then be executed to split these files, resulting in a set of split restart files for use with the split input deck (with restart flags set appropriately to utilize the restart files). The input file required by this utility must be named **restart_split.inp**, and contain the following information:

```
# input:  restart file(s) to be split
RESTART_IN     restart

# output: split restart file(s)
RESTART_OUT    restart_split
```

where RESTART_IN specifies the root name of the restart file to be split, and RESTART_OUT specifies the root name of the split restart files that are output. As is the case for all of the input decks associated with SGLD executables, blank lines are ignored and comment lines are indicated by "#". This utility can currently only split restart data files for regions that are solved in an elliptic fashion with VULCAN.

Chapter 4

Merging Restart and Post-Processed Files

The **restart_merge** utility of SGLD provides the capability to merge the split restart files back to their original non-split form. This capability is useful if one desires at some point to load balance for a different number of processors than what was originally considered. The following is a sample input file (called **restart_merge.inp**) that is required by this utility:

```
# input:  restart file(s) to be merged
RESTART_IN     restart_split

# output: merged restart file(s)
RESTART_OUT    restart_merge
```

where RESTART_IN specifies the root name of the restart files to be merged, and RESTART_OUT specifies the root name of the merged restart files that are output. Blank lines are ignored, and comment lines are indicated by "#". This utility can currently only merge restart data files for regions that are solved in an elliptic fashion with VULCAN. The PLOT3D files created by the VULCAN post-processor are merged by executing the PLOT3D merge utility, **plot3d_merge**. This utility does not utilize an input deck. Instead, this utility requires that the PLOT3D data files be located in the Plot3d_files directory of the directory it is executed from. Hence, this utility should be executed from within the working directory of a given VULCAN simulation. Note that the **plot3d.g.fvbnd** file, which contains the information required to display data at external boundaries, is not currently included as part of the merge process. If this file is desired, the VULCAN post-processor must be executed one time using the original non-split VULCAN input deck. This will create the **plot3d.g.fvbnd** which can be saved for future use.

By default, the merge process will be performed as the last step of the VULCAN execution process if the file MERGE_MAP.DAT (created when executing **grid_split**) is present in the working directory of a VULCAN simulation. If merging of the restart and/or PLOT3D files is not desired, simply rename this file to something other than MERGE_MAP.DAT.